PROMAL

PROgrammer's Micro Application Language

QUESTIONS AND ANSWERS ABOUT PROMAL

As programmers of microcomputers, we need all the help we can get. PROMAL is designed to give IBM, Apple and Commodore programmers an efficient, productive development language and environment. I hope these questions and answers will help you decide that PROMAL is your *best next language.*

John R. Segner
President,
SMA, Inc.

WHAT IS PROMAL AND WHO IS IT FOR?

PROMAL stands for **PRO**grammer's **M**icro **A**pplication **L**anguage. It's a **new programming system.** It's a high-level **structured** programming language. It includes a fast one-pass **COMPILER,** a full-screen **EDITOR,** a command **EXECUTIVE,** and a **LIBRARY** of pre-defined utility subroutines. But who should use PROMAL?

PROMAL is for programmers: BASIC *programmers* who want to move up to a high-level, but high-performance language. It's also for *intermediate programmers* who want a less complex alternative to Assembler or an easier, more efficient language than Pascal. And, it's for *advanced programmers* who want to enjoy the full potential of their computers with a complete development system and a high-performance, advanced compiled language that can easily interface to machine language.

WHAT DO I GET WITH PROMAL?

First PROMAL is a *language and compiler.* It is a new structured programming language, designed from "scratch" to run efficiently on microcomputers. A fast, one-pass compiler compiles the language to object code which, for Apple and Commodore machines, is executed by the run-time nucleus of the PROMAL Executive. For IBM machines, "native" object code is generated by the compiler.

The PROMAL *Executive* is an "operating system" environment which has powerful commands for loading and executing programs (with multiple programs in memory at once), for managing files and directories, for displaying and changing memory, for I/O re-direction, and for executing "batch" JOB files.

Next is the *Library,* which consists of machine-language subroutines called by PROMAL source statements. These are fast and efficient routines for handling all kinds of input and

output, for string operations, editing, cursor control, managing files, and much more.

Finally there is the full-screen *Editor.* It supports insert/delete, search/replace, block moves/copies, save to file/copy from file, indent/un-indent, function key commands with on-screen prompts, a help key, and an "edit-after-error" facility. Edit-after-error means that the Editor can locate compile errors and position the cursor to the statement in error after a compilation attempt.

WHAT ARE THE ADVANTAGES OF PROMAL?

Advantage #1: Performance

The PROMAL compiled language gives you *very high performance* for graphics, games, text-processing applications, music synthesis and business applications. PROMAL programs will run up to 2000% (or more) faster than BASIC programs. (Benchmark results are given later in this booklet.) Now you can write application programs, in a powerful high-level language, which previously could only be written in tedious machine codes.

Advantage #2: Simplicity

PROMAL is *easier to learn and use* than Pascal or "C" or FORTH. Here's why: There are fewer "rules" to follow. Blanks are the only separators and there are no fussy terminators like ";" or "}" or "." to worry abut. You only put one statement per line (except when defining DATA constants) and there are no line numbers. Comments take no memory and can be used freely to improve readability. B*ut most important of all*: there are no END statements to match up with IFs, WHILEs and other statements. PROMAL uses *indentation* as a syntax element to create a simple structured programming style.

With PROMAL you will write programs that are easier to read and understand than you have ever written before.

Advantage #3: Efficiency

With the PROMAL system you get an *efficient* development environment. You enter and modify programs easily with the Editor; you compile from and to memory "workspace" and execute from the Executive. Your *development cycle is shortened* because your program can be quickly modified, recompiled and tested.

You also get an "edit after error" facility. Any *compilation errors are located* in the source program by the Editor and the cursor is positioned to the error automatically, so you can quickly correct the error and re-compile.

Advantage #4: Productivity

The PROMAL Executive has *many powerful commands* which give you *"operating system"* control of your computer. You get commands for file and directory management, memory maps, memory display and change, multiple programs in memory, load/unload of programs, JOB files, and I/O redirection. Each of these can help you be more productive in your programming.

The productivity, power and sophistication of PROMAL are fact, not fantasy; the PROMAL Compiler, the Editor and the Executive *are written entirely in* PROMAL! (If you've ever tried writing a full-screen editor in BASIC or Pascal then you'll really appreciate PROMAL.)

WHAT IS THE PROMAL SYSTEM, EXACTLY?

First, there's the PROMAL Language and its Compiler

PROMAL is an entirely *new* structured programming language. It provides the control statements and structures known in computer science as "structured constructs." As you may know, the constructs are "sequence", "if-else", "do-while", "do-until" and "case". Procedures, subroutines and functions are structures also commonly considered required for structured programming.

PROMAL implements structured programming through control statements, indentation, and its subroutine capabilities. Its control statements include: IF, IF-ELSE, WHILE, REPEAT (a "do-until"), FOR, BREAK, NEXT, and CHOOSE (a "case" construct). Procedures, named subroutines, and functions (all with passed arguments) are fully supported.

But it has a truly unique "structuring" feature. . .

PROMAL uses "indentation" as a syntax element of its control statements. Statements after IF, WHILE, REPEAT, FOR, and CHOOSE must be indented two spaces. The control statements are ended by the next "un-indented" statement. (There are no "END" statements or semi-colons required in PROMAL to terminate these statements.)

Look at the difference between two functionally identical code segments. The first is written in Commodore BASIC and the second is in PROMAL:

```
1210 POKE(53281),5
1220 IF PEEK(53278) AND PEEK(53249) THEN E=A:GOSUB2750:S=S+1
```

Now here is the same code in PROMAL:

```
IF collision AND alien  ; check for collision
  EXPLODE alien         ; and call subroutine
  score = score + 1     ; increment score
```

With PROMAL the code is easy to read and understand. Comments can be used freely since they take no memory space. Also, PROMAL is designed to help you maintain good programming habits and style.

All variables are declared before use (so you never have any surprises later) and they can be up to 31 characters long — so now you can have truly meaningful names. All procedures and functions must be defined before they are used, which helps prevent confusion later. You exit from loops with BREAK and NEXT. And you can even exit "gracefully" from lower-level subroutines with the "ESCAPE" statement.

The *compiler* is a one-pass, recursive descent compiler, designed to generate a highly compact and efficient object code. (More on this later).

Compliation is fast. . .you can compile a 100 line program typically in less than 10 seconds! And the object code generated is very compact — on the average, source statements compile down to about 6 bytes each of object code!

Next is the "Executive".

The **Executive** is a control program (like an "operating system") which lets you load and execute programs, manage your disk files (copy, rename, delete), display and change memory, invoke the compiler and editor, and much more.

For example, you get a powerful "I/O redirection" capability similar to that found in UNIX. You can also define up to 8 memory resident commands to customize the Executive, and you can define any number of disk-based commands.

The Executive has features you would only expect to find in "big system" programming environments.

Then there is the "Editor".

The **Editor** is like some of the better word processors available today, but it was specifically designed for PROMAL program development (although it can be used as a word processor for text or for creating data files).

It has *full-screen* editing with insert and delete, search and replace, block move and copy to and from files as well as within a program. It has indent and undent commands to support PROMAL's indentation. Function keys are used to execute commands and a help screen is always available.

The editor can edit and save to a memory "workspace" for rapid changes and re-compilations in memory. You'll like how easy and simple it is to enter and edit code when you start using the Editor to write your program.

Finally there is the "Library".

The **Library** is a set of 45 machine-language subroutines which are called from PROMAL source statements. They are fast and efficient routines to do various types of INPUT and OUTPUT, manage files, do STRING operations, convert from one data type to another for formatted output, and much more. Library routines are always memory-resident to make execution lightning-fast.

WHAT ARE SOME TECHNICAL DETAILS AND SPECIFICATIONS?

About the Compiler: The PROMAL compiler generates a very compact object code which, for 6502 systems, is executed by a machine-language nucleus in the Executive. It executes faster than other products which produce native 6502 code, because the nucleus routines have been carefully optimized for maximum execution speed. In fact, a year of intense research into 6502 code generation techniques preceded the PROMAL compiler development.

For 808X-based machines the compiler produces true "native" object code. Execution can be under the control of the Executive or as stand-alone programs.

Machine language subroutines can be called (with passed arguments) from any PROMAL program as normal subroutines, embedded "inline" as object code using DATA statements, or dynamically loaded from disk with the MLGET function. Machine language subroutines may also be loaded from the Executive with a GET command, or loaded through a batch job.

Real numbers are supported to 11 places of precision — 2 more than BASIC. And the floating-point support is extremely accurate. We ran David Ahl's benchmark (Ahl's Simple Benchmark) as published in the July 1984 issue of *Creative Computing* and found that PROMAL's floating-point arithmetic was *more accurate* than 148 of the 196 systems tested (including some multi-million dollar "mainframe" computers).

And PROMAL has bit-level operators, supports hex numbers, WORD and BYTE data types, and has other capabilities (like pointers) for handling difficult programming tasks which normally would require machine language.

The language statements include:

```
PROGRAM, PROC, FUNC     (procedure definition)
DATA, CON               (constant definition)
BYTE, WORD, INT, REAL   (variable declaration)
=                       (assignment)
IF. . .                 (If. . .Then conditional)
IF. . .ELSE. . .        (If. . .Then. . .Else conditional)
WHILE. . .              (DO While)
REPEAT. . .UNTIL        (DO Until)
FOR. . .TO              (indexed loop)
CHOOSE. . .ELSE         (CASE statement)
BREAK                   (loop exit)
NEXT                    (iteration control)
NOTHING                 (null statement)
ESCAPE                  (subroutine escape)
REFUGE                  (escape-to code)
RETURN                  (subroutine function return)
```

About the Editor: The editor is a full-screen cursor-driven design and supports 80-character lines. Commands are executed by control and function keys and the current function key definitions are always shown at the bottom of the screen. Of course, there's line and full-screen scrolling. And, like the more advanced word processors, a "highlighting" technique is used for defining block operations.

You get all the edit features you'd expect from a good program editor: line insert, delete, search; block copy, move, delete; string search and replace; and block write to/read from file. Plus, there's auto indent and undent for PROMAL statement structuring.

The editor uses a memory resident edit-buffer architecture. With an average statement length of 24 characters you could edit a program of about 1000 source lines on a 64K Commodore 64. But you can compile much larger source programs. That's possible using the compiler's powerful "include" feature.

The INCLUDE statement allows procedures or blocks of statements to be included from other files on disk into your source program at compile time. This means that the memory available for the Editor *does not limit the size* of a program that can be compiled with PROMAL.

About the Executive: The Executive gives you so much function and power you will wonder how you ever managed without it. Here are the highlights. You get:

- Program execution by command name
- Multiple programs in memory at once
- Type any file to screen or printer
- Memory map; display and change memory capability
- File management (copy, rename, delete) and directory display
- Function key redefinition
- Batch JOB files
- I/O Redirection

Batch JOBs and I/O Redirection are powerful "operating system" features which require more explanation unless you are familiar with UNIX or MS-DOS.

The Executive lets you *execute a "JOB" file.* This file can contain any PROMAL commands or program names. The Executive reads the JOB file and executes the commands as if they had been entered one after the other from the keyboard.

You can automate elaborate procedures, chain program execution, execute another JOB file, or even re-execute the same JOB file in an endless loop.

With the Executive's I/O *Redirection* function you can "redirect" input and output to and from the various I/O devices. For example, you could redirect a disk file as keyboard input for a program. Or, you could "type" a file to disk or to the printer by redirecting the screen display output.

Here's how I/O redirection works. Suppose you wanted a quick printout of a diskette directory. All you have to do is type: "FILES >P". The FILES command normally displays the directory on the screen, but the output is redirected to the printer by the " > P" command. The directory display is sent to the printer!

There are 24 pre-defined Executive commands:

Command	Function
COLOR	Change screen color
COPY	Copy a file
CS	Clear screen
DATE	Set or change date
DELETE	Delete a file
DUMP	Display memory (Hex & ASCII)
EDIT	Invoke the Editor
FILES	Display directory information
FILL	Fill a memory area
FKEY	Redefine a function key
GET	Load PROMAL or Machine Language program
GO	Execute machine language program
HELP	Display Help screen
JOB	Execute job command file
MAP	Display memory map
NOREAL	Unload real number support
PAUSE	Pause JOB file execution
QUIT	Exit PROMAL system
RENAME	Rename a file
SET	Set memory locations
SIZE	Display compiled program size
TYPE	Display a file
UNLOAD	Remove a program from memory
WS	Clear or alter ''Work-space'' size

The PROMAL Executive gives you real "operating system" power. Plus, you can develop your own commands to customize and extend the Executive. Note: For the MS-DOS version, the Executive's file management commands, where duplicated by MS-DOS commands, are not implemented.

About the Library: The *Library* provides high-speed machine-language subroutines which are called by PROMAL source statements for frequently needed language tasks, such as string handling, I/O operations, editing, type conversions, cursor control, file handling and other functions. By keeping these facilities separate, but callable from the language, important savings in memory and increases in speed of execution can be achieved.

The memory-resident LIBRARY routines are:

Name	Decription
ABORT	Abort program execution
APLHA	Test if character is alphabetic
BLKMOV	Block move
CHKSUM	Compute 16 bit checksum of region
CLOSE	Close an open file or device
CMPSTR	Compare strings
CURCOL	Determine current cursor column
CURLINE	Determine current cursor line
CURSET	Position cursor on the display
DIR	Display file names matching a pattern
EDLINE	Edit a line on the screen.
EXIT	Exit from program, w/ optional message
FILL	Fill a memory block with a constant
FKEYGET	Get the current function key definition
FKEYSET	Define a function key expansion string
GETARGS	Split command line into agruments
GETBLKF	Block-read from file
GETC	Input one character from the keyboard
GETCF	Input one char. from file or device
GETL	Input one line from the keyboard
GETLF	Input one line from a file or device
INLINE	Input a line with screen editing
INSET	Test if a character is in a string
INTSTR	Convert signed value to a string
LENSTR	Return length of string
LOOKSTR	Search a list of strings
MAX	Return the largest of arguments
MIN	Return the smallest of arguments
MOVSTR	Copy, substring or concatenate strings
NUMERIC	Test if character is numeric
OPEN	Open a file or device for input/output
OUTPUT	Formatted output with many options
OUTPUTF	Formatted output to a file
PUT	Output text to the display
PUTBLKF	Block-write to file
PUTF	Output text to a file or device
RANDOM	Obtain a pseudo-random number
REALSTR	Convert a REAL value to a string
RENAME	Rename a file
STRREAL	Convert string to a REAL numeric value
STRVAL	Convert string to a numeric value
TESTKEY	Test if a key is pressed on keyboard
TOUPPER	Fold lowercase letter to uppercase
WORDSTR	Convert an unsigned value to a string
ZAPFILE	Delete a file

WHAT ARE THE LIMITATIONS OF PROMAL?

To achieve our speed and efficiency goals for PROMAL we had to streamline it by trimming off some extra "weight"

First, we don't have the myriad of complex data-structure definitions of Pascal. But our BYTE data type lets you define any data-structure you need.

Second, at Release level 1.1, separately compiled modules are not supported. However, a future release will provide such support. Currently, users can achieve programming modularity through the compiler's INCLUDE capability.

But, these tradeoffs are worth it when you look at PROMAL's speed. . .

PROMAL BENCHMARK RESULTS

The tables below summarize our benchmark tests for the Commodore 64. The test program is the standard "Sieve of Eratosthenes" benchmark test from BYTE Magazine (September 1981) configured to generate prime numbers between 3 and 3600.

Company	Product	Time (secs.)
SMA, Inc.	**PROMAL 1.1**	**30**
Micro Products	FORTH	51
Abacus Software	ZOOM PASCAL (native 6502 code)	55
Skyles Electric	BLITZ! BASIC Compiler	154
Limbic Systems	OXFORD PASCAL	182
CodeWriter Corp.	SPEED WRITER BASIC Compiler	201
Kyan Software	KYAN PASCAL Version 2.0	241
COMAL Users Gp.	COMAL 0.14	490
Commodore	COMMODORE BASIC	630

This table summarizes some additional benchmark tests:

	PROMAL	BASIC	COMAL	FORTH	PASCAL
Execution time (secs.)	**30**	630	490	51	55
Object code size (bytes)	**128**	255	329	181	415
Program load time (secs.)	**3.2**	3.8	6.3	11.2	23.5
Compile time (secs.)	**8.5**	NA	NA	3.9	108

The products used in the above tests were: Commodore BASIC, COMAL User's Group USA COMAL, Micro Products FORTH, and Abacus ZOOM Pascal. The full benchmark report (available from SMA) contains details on how the above tests were made and definitions of the performance measures.

WHAT CAN I DO WITH PROMAL?

Any type programming is possible with PROMAL. Because it has certain features which are normally used in machine-language programming (pointers, byte data types, bit-level operators, hex numbers, external variable declarations) PROMAL can be used instead of machine language for most applications requiring high performance such as graphics, games, text-processing, and communications programming.

In addition to traditional "applications" programming, PROMAL is suitable as a "systems" programming language for developing compliers, assemblers, editors, or real-time control systems. The following chart summarizes PROMAL against BASIC and PASCAL for various types of programming.

Application	PROMAL	BASIC	PASCAL
Communications	Yes	No	Maybe
Text-processing	Yes	Maybe	Maybe
Games, Animation	Yes	No	No
Compilers	Yes	No	Maybe
Assemblers	Yes	No	Maybe
Music Synthesis	Yes	No	No
Business applications	Yes	Yes	Yes
Graphics (hi-res)	Yes	Yes	Yes
Real-time Control	Yes	No	No

HOW IS PROMAL DOCUMENTED?

We have tried to make PROMAL the best-documented system ever in its price range. There are three manuals bound together in an attractive vinyl 3-ring binder — over 230 pages — complete with a detailed index.

First, there's a simple introductory manual called "MEET PROMAL!" which gets you started. It takes you step by step through each system component. Then there is a complete USER'S GUIDE which explains the Executive, the Editor and the Compiler. It's full of easy-to-follow examples.

The third manual is the PROMAL LANGUAGE MANUAL. It completely documents the language and has examples of every statement and language element. Everything you need to learn the PROMAL language is covered, in an easy-to-read style and logical presentation. Additional technical topics which don't fit easily into the main text are found in the 15 Appendices to the manual.

WHAT ARE THE ADVANCED FEATURES OF PROMAL?

The PROMAL system gives the programmer some very advanced features, some of which are only found in much larger operating systems (UNIX, MS-DOS).

Advanced language features:

- Recursive subroutines supported
- INCLUDE files at compile time
- Direct access to machine language DOS & "kernal" routines
- Inline machine language code definition via DATA statements
- Pointers to all data types; cast operators for type conversion
- Bit-level operations: BYTE, WORD, INT, REAL data types; hex numbers
- Formatted numeric output (for example:"$167.00")
- 11-place precision for floating point arithmetic
- External variable definition

Advanced Editor features:

- Block definition high-lighting
- Auto indent, un-indent support
- Save Block to file, Load Block from file
- Search/replace with confirm
- 80 column lines on 40 column screen

Advance Library features:

- File directory, rename, delete routines
- Line editing routine
- Advanced string handling routines
- Formatted I/O support
- STDIN, STDOUT for I/O redirection within a program

Advanced Executive features:

- Memory map display
- Display & change memory
- I/O Redirection commands
- Memory workspace supported as a device
- Execute file of commands (JOB file)
- Extensible with 8 additional memory-resident commands, unlimited disk commands
- GET command for program load
- JOB command for executing JOB files

WHAT DO I GAIN FROM USING PROMAL?

1. *High performance from a structured language.* — The PROMAL language is fast, powerful and easy to use.

2. *An efficient and productive development environment* — The operating system Executive gives you *file management, memory management,* I/O *redirection, program management,* and other useful *utility* functions.

3. A *rapid way to write source code* — The full-screen, cursor-driven Editor lets you quickly enter and edit source programs. (And it even locates your compilation errors.)

4. *Time-saving subroutines* — The Library of machine language subroutines supports the run-time environment with optimized routines for file I/O, string handling, formatted output, cursor control, and data conversion, etc.

5. *Ease of learning and use* — Clearly written, comprehensive tutorial and reference manuals make PROMAL easy to learn and to use.

WHAT'S NEW IN VERSION 1.1?

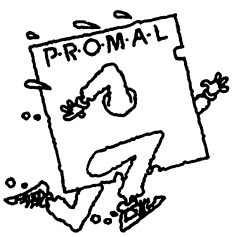
We've added some terrific features to Version 1.1 which include:

- Trigonometric log, square root, exponential and power functions
- Support for RS-232 devices and a sample "terminal emulator" program which users many modify and extend
- Support for Commodore 64 "relative" files and a sample PROMAL application program for creating and accessing relative files
- An advanced machine language subroutine interface for directly accessing DOS or "kernal" routines
- A Diskette Utility program for formatting and copying diskettes (Commodore and Apple versions only)
- Non-copy protected master diskette for backup convenience

And we're now offering PROMAL in two configurations. First, we have the "End-user" package which is for development and execution on a single system and uses the Executive's run-time support. Second, is the "Developer's" package which includes an object-module generation utility which binds the run-time nucleus with your application program so that it can run on any system without the PROMAL Executive. Included with this package is an unlimited license to sell or distribute your PROMAL application programs.

Supported Computers:

Commodore 64 and 128
Apple IIe with 80 column card & PRODOS
Apple IIc
IBM PC, PC-Jr, XT, AT (call for availability)



WHAT DOES PROMAL COST?

The complete PROMAL "End-User" system, with manuals, 3-ring binder and PROMAL System and Demo Diskettes including sample programs is *only* $49.95 (plus $5.00 shipping and handling if ordered direct from SMA, Inc.) 100% credit of your purchase price is allowed toward later purchase of the "Developer's Version". There is a 15-day, no-risk, moneyback guarantee. If you're writing programs just for yourself to run on your system, this is all you need.

The Developer's Version is *only* $99.95 (plus $5.00 S&H if ordered direct). You receive all the above plus an unlimited right to sell or distribute PROMAL applications. You get the "run time" **object module** generation capability and extra documentation — plus everything in the End-User system —and you get the guarantee. If you are a software developer... this is what you want.

WHAT DO PROMAL USERS SAY ABOUT PROMAL?

Here's what come of our first users (that started with Version 1.0) have said in unsolicited letters we've received:

"*Excellent is the best way to describe your* PROMAL *system. It has taken the best of* UNIX *and* C *to produce an ideal development system. . . Well done indeed!*"

M. T. V.
Naperville, Ill.

This note was included with a registration card. It said in part:

"*I am. . .so amazed by* PROMAL *that I have to include this note. . .I cannot believe the high degree of excellence of this entire package.*"

C. P., Ph.D.
Ridgeway, NY

Edward M. Rowe is an independent reviewer and freelance writer in Randolph, New Jersey. He reviewed PROMAL in the April 1985 issue of *Run* magazine and concluded:

"*. . .I'm very impressed with* PROMAL. *I find the programming environment friendly and a welcome change from* BASIC. *The programs are compiled and executed extremely fast. The documentation is professional and consists of a* 200 *page manual. . .I believe that* PROMAL *will be a powerful addition to your Commodore* 64."

Edmund C. Rowan of Rowan Computer Services in Alexandria, Virginia wrote:

"*I don't know that I've ever seen a [system] as thoughtfully designed and as skillfully executed as* PROMAL. *Its logic and ease of programming are truly remarkable. Its speed of execution is phenomenal. . .congratulations on* PROMAL!"

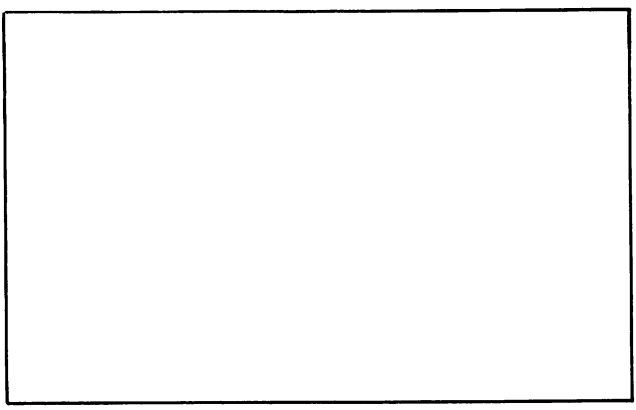
CAN PROMAL BE SUMMED UP IN ONE SHORT PARAGRAPH?

Yes. PROMAL is a totally new *programming development system* which provides optimum productivity and performance for writing most types of microcomputer applications. The compiled language is a new structured programming language which combines the best features of languages like "C" and PASCAL without the complexity and learning curve of either. The EXECUTIVE (operating system) and EDITOR (plus the LIBRARY) integrate to form the ultimate environment for programming microcomputer applications.

WHERE CAN PROMAL BE PURCHASED?

PROMAL is available direct from SMA, Inc. or from your local computer dealer.

Your local PROMAL Dealer is:



PROMAL is a trademark of:

Systems Management Associates, Inc.
3325 Executive Drive
Raleigh, North Carolina 27619
(919) 787-7703

Copyright (C) 1985, SMA, Inc. All Rights Reserved

PROMAL specifications and pricing are subject to change without notice.